Constraint-based measure for estimating overlap in clustering

Antoine ADAM ANTOINE.ADAM@CS.KULEUVEN.BE
Hendrik Blockeel HENDRIK.BLOCKEEL@CS.KULEUVEN.BE
KU Leuven, Department of Computer Science, Celestijnenlaan 200A, 3001 Leuven, Belgium

Keywords: meta-learning, constraints, clustering

Abstract

Different clustering algorithms have different strengths and weaknesses. Given a dataset and a clustering task, it is up to the user to choose the most suitable clustering algorithm. In this paper, we study to what extent this choice can be supported by a measure of overlap among clusters. We propose a concrete, efficiently computable constraint-based measure. We show that the measure is indeed informative: on the basis of this measure alone, one can make better decisions about which clustering algorithm to use. However, when combined with other features of the input dataset, such as dimensionality, it seems that the proposed measure does not provide useful additional information.

1. Introduction

For many types of machine learning tasks, such as supervised learning, clustering, and so on, a variety of methods is available. It is often difficult to say in advance which method will work best in a particular case; this depends on properties of the dataset, the target function, and the quality criteria one is interested in. The research field called meta-learning is concerned with devising automatic ways of determining the most suitable algorithm and parameter settings, given a particular dataset and possibly knowledge about the target function. Traditionally, meta-learning has mostly been studied in a classification setting. In this paper, however, we focus on clustering.

Clustering algorithms are no exception to the general rule that different learning algorithms make different assumptions about the input data and the target function to be approximated. For instance, some clustering algorithms implicitly assume that clusters are spherical; k-means is an example of that. Any clustering algorithm that tries to minimise the sum of squared Euclidean distances inside the clusters, implicitly makes that assumption. The assumption can be relaxed by rescaling the different dimensions or using a Mahalanobis distance; this can lead to elliptic clusters, but such clusters are still convex.

A different class of clustering algorithms does not assume convexity, but looks at local properties of the dataset, such as density of point or graph connectivity. Such methods can identify, for instance, moon-shaped clusters, which k-means cannot. Spectral clustering (von Luxburg, 2007) is an example of such approach.

Some clustering algorithms assume that the data have been sampled from a population that consists of a mix of different subpopulations, e.g., a mixture of Gaussians. EM is an example of such an approach (Dempster et al., 1977). A particular property of these approaches is that clusters may overlap. That is, even though each individual instance still belongs to one cluster, there are areas in the instance space where two (or more) Gaussian density functions substantially differ from zero, so that instance of both clusters may end up in this area.

In this paper, we hypothesise that the amount to which clusters may overlap is relevant for the choice of what clustering method to use. A measure, the Rvalue, has been proposed before that, given the ground truth regarding which instance belongs to which cluster, describes this overlap. Since clustering is unsupervised, this measure cannot be used in practice for deciding what clustering method to use. We therefore derive a new measure, CBO, which is based on must-link or cannot-link constraints on instance pairs. We show that the second measure correlates well with the first, making it a suitable proxy for selection the clustering

Preliminary work. Under review for Benelearn 2017. Do not distribute.

method. We show that this measure is indeed informative: on the basis of this measure alone, it is possible to select clustering algorithms such that, on average, better clusterings are obtained.

However, there are also negative results. Datasets can be described using other features than the measure defined here. It turns out that, when a dataset is described using a relatively small set of straightforward features (such as dimensionality), it is also possible to make an informed choice about what clustering method to use. What's more, if this set of straightforward features is extended with the overlap measure described here, this does not significantly improve the informativeness of the dataset description, in terms of which clustering method is optimal.

The conclusion from this is that, although the proposed measure is by itself an interesting feature, it seems to capture mostly information that is also contained in other, simpler features. This is a somewhat surprising result for which we currently have no explanation; further research is warranted.

This paper is the continuation of a previously published workshop paper (Adam & Blockeel, 2015). While following the same ideas, the CBO has been completely redefined. In addition, the number of datasets considered was increased from 14 to 42. While the correlation of the CBO with the overlapping has improved considerably, the promising results for the algorithm selection of that paper were somewhat reduced by adding those datasets.

The remainder of this paper is structured as follows. Section 2 discusses some related work on constraint-based clustering and meta-learning for clustering. Section 3 studies how the overlapping of clusters influences the performance of algorithms. Section 4 introduces CBO, which is intended to approximate the amount of overlap from constraints. Section 5 presents experimental results that compare algorithm selection based on CBO with algorithm selection using other features of the dataset. Section 6 presents our conclusions.

2. Related work

2.1. Constraint-based clustering

Clustering is the unsupervised learning task of identifying groups of similar instances in a dataset. Although these groups are initially unknown, some information can be available as to what the desired solution is. This information takes the form of constraints on the resulting clusters. These constraints can be provided to the clustering algorithm to guide the search towards a more desirable solution. We then talk about constraint-based, constrained, or semi-supervised clustering.

Constraints can be defined on different levels. On a cluster level, one can ask for clusters that are balanced in size, or that have a maximum diameter in space. On an instance level, one might know some partial labelling of the data. A well-used type of constraints are must-link and cannot-link constraints, also called equivalence constraints. These are pair-wise constraints which state that two instances must be or cannot be in the same cluster.

Multiple methods have been developed to use these constraints, some of which are mentioned below. A metric can be learnt that complies with the constraints (Bar-Hillel et al., 2005). The constraints can be used in the algorithm for the cluster assignment in a hard (Wagstaff et al., 2001) or soft way (Pelleg & Baras, 2007), (Ruiz et al., 2007), (Wang & Davidson, 2010). Some hybrid algorithms use constraints for both metric learning and clustering (Bilenko et al., 2004), (Hu et al., 2013). Other approaches include constraints in general solver methods like constraint programming (Duong et al., 2015) or integer linear programming (Babaki et al., 2014).

2.2. Algorithm selection for clustering

Little research has been conducted on algorithm selection for clustering. Existing methods usually predict the ranking of clustering algorithms (De Souto et al., 2008), (Soares et al., 2009), (Prudêncio et al., 2011) (Ferrari & de Castro, 2015). The meta-features used are unsupervised and/or domain-specific. None of these approaches are using constraints which removes the specificity that there is not only one single clustering for one dataset. To the best of our knowledge, the only meta-learning method for clustering involving constraints is (Van Craenendonck & Blockeel, 2016) which does not use features but simply selects the algorithm that satisfies the most constraints.

3. Rvalue

As already mentioned, we assume some algorithms can handle overlapping better than others. For example, figure 1 shows a toy dataset (on the left) where two Gaussians overlap in there centre, forming a cross. In that case, EM (in the middle) is capable of retrieving the correct clustering while spectral clustering (SC, on the right) cannot. This shows the relevance of overlapping as a meta-feature to select a clustering algorithm.

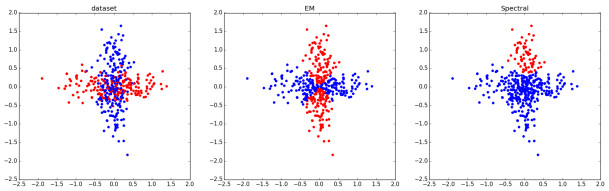


Figure 1. Toy example of the cross dataset.

The Rvalue (Oh, 2011) has been used before as a measure of overlapping. Given a dataset of instances in different classes, it quantifies the overlapping as a number between 0 and 1. To compute the Rvalue of a dataset, it considers each instance and its neighbourhood. An instance is said in *in overlapping* if too many of it nearest neighbours are labelled differently than him. The Rvalue of a dataset is then the proportion of instances *in overlapping*. The Rvalue thus has 2 parameters: the k-nearest neighbours to consider, and θ, the number of nearest neighbours from a different class above which an instance is *in overlapping*. Figure 2 shows the Rvalue for some UCI datasets, which shows overlapping occurs a lot in real-life datasets. For comparison, the cross dataset just above has an Rvalue of 0.41 for the same parameters.

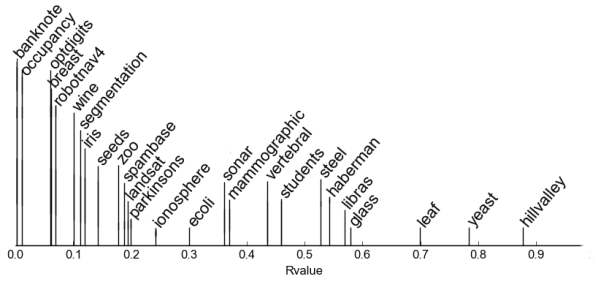


Figure 2. Rvalue of some UCI datasets, $k = 6$, $\theta = 1$.

To check our intuition that EM can handle overlapping better than SC, we look at the performance of these algorithm w.r.t. the Rvalue. Table 1 shows the average performance of these algorithm over some UCI datasets presented in further sections. Table 2 shows the same results but ignoring datasets were both algorithm performed badly. We assume that if both algorithm have an Adjusted Rand Index (Hubert & Arabie, 1985) (ARI) of less than 0.2, the dataset is not very suitable for clustering to begin with and we can then ignore it. A complete list of used datasets can be found in section 4.3. It can be seen that in that second case, EM performs better than SC when there is overlapping and vice versa when there is no or little overlapping. This difference is much reduced when including bad performing datasets. This suggest strongly that while overlapping does impact some algorithms more than others, other factors also have a significant influence of the performance of clustering algorithms.

	EM	SC
all	0.31	0.32
$Rvalue < 0.2$	0.48	**0.50**
$Rvalue > 0.2$	**0.19**	**0.19**

Table 1. Average clustering performance measured with ARI.

	EM	SC
all	0.45	0.47
$Rvalue < 0.2$	0.55	**0.59**
$Rvalue > 0.2$	**0.33**	0.31

Table 2. Same as table 1 for dataset where either EM or SC scored an ARI of at least 0.2.

4. Detecting overlap using constraints

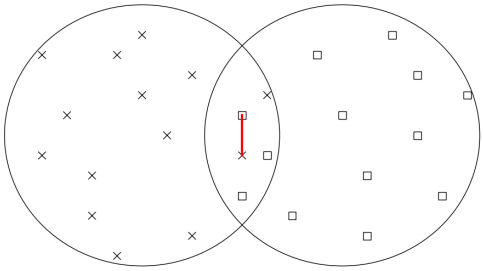
While the Rvalue is a good indicator of the extent to which clusters overlap, it is not useful in practice because it requires knowledge of the clusters, which we do not have. In this section, we present an alternative measure: the Constraint-Based Overlap value (CBO). The CBO is designed to correlate well with the Rvalue, while not requiring full knowledge of the clusters.

4.1. Definition

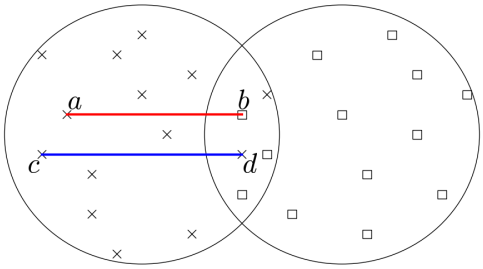
The CBO makes use of must-link and cannot-link constraints. The idea is to identify specific configurations of ML or CL constraints that indicate overlap. The CBO uses two configurations, illustrated in figure 3:

- short CL constraints: when two points are close together and yet belong to different clusters, this is an indication that the two clusters overlap in this area
- two parallel constraints, one of which is ML and the other CL, between points that are close. That is, if a and c are close to each other, and so are b and d, and a and b must link while c and d cannot link, then this implies overlapping, either around a and c or around b and d (see figure).

The more frequent those patterns, the more the clusters overlap. A limit case of the second configuration is when the 2 constraints involves the same point (e.g. $a = c$ on the figure) Then, by propagation of



(a) Short cannot-link pattern



(b) Parallel and close must-link and cannot-link pattern

Figure 3. Overlapping patterns in constraints. The crosses cluster and the squares cluster, both represented by a circle, overlap in the middle. A red line signifies a cannot-link, while a blue line signifies a must-link constraint.

the constraints, there is a short cannot-link constraint between the other 2 points.

The question is how to define "short" or "close". This has to be relative to "typical" distances. To achieve this, we introduce a kind of relative similarity measure, as follows. Let $d(x, x')$ be the distance between points x and x', and ϵ (ϵ') be the distance between x (x') and its k'th nearest neighbour. Then

$$s(x,x') = \begin{cases} 1 - \frac{d(x,x')}{max(\epsilon,\epsilon')} & if d(x,x') \leq max(\epsilon,\epsilon') \\ 0 & otherwise \end{cases}$$

That is: $s(x, x')$ is 1 when x and x' coincide, and linearly goes to 0, reaching 0 when $d(x, x') = \max(\epsilon, \epsilon')$, that is, x is no closer to x' than its k'th nearest neighbour, and vice versa.

Using this relative similarity, we can assign scores to both types of configurations mentioned above.

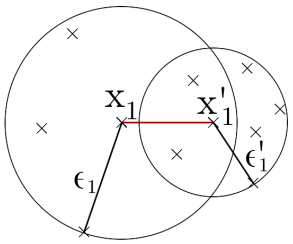
The score of a **short constraint** between two points x and x' is simply:

$$score(c) = s(x, x')$$

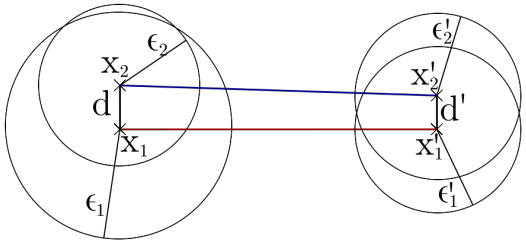
The score for a **pair of parallel constraints**, c_1 between points x_1 and x_1' and c_2 between x_2 and x_2', is computed as follows. Without loss of generality, assume $d(x_1, x_2) + d(x_1', x_2') \leq d(x_1, x_2') + d(x_1', x_2)$ (this can always be achieved by renaming x_2 to x_2' and vice versa, see figure 4(b)). We then define:

$$score(c_1, c_2) = s(x_1, x_2) \times s(x_1', x_2')$$

The multiplication ensures that if either x_1 and x_2 or x_1' and x_2' are too far apart then the score is zero.



(a) Score of a single constraint



(b) Score of a pair of constraints

Figure 4. Scoring of single constraint(a) and pair of constraints(b) using the local similarity. The circles represent the neighbourhoods of the points.

In both cases, higher scores are more indicative of overlap. To have a measure for the whole dataset, we aggregate these scores over the whole constraint set. The idea is to compare the amount of short cannot-link constraints, direct (single pattern) or by propagation(double pattern), to the total amount of short constraints, both must-link and cannot-link. With CL the set of cannot-link constraints and ML the set of must-link constraints, we define

$$CBO = \frac{\sum_{c \in CL} score(c) + \sum_{\substack{c_1 \in CL \\ c_2 \in ML}} score(c_1, c_2)}{\sum_{c \in CL \cup ML} score(c) + \sum_{\substack{c_1 \in ML \\ c_2 \in CL \cup ML}} score(c_1, c_2)}$$

4.2. Stability

As one can imagine, the CBO can be very noisy for very small constraint sets. Several parameters influ-

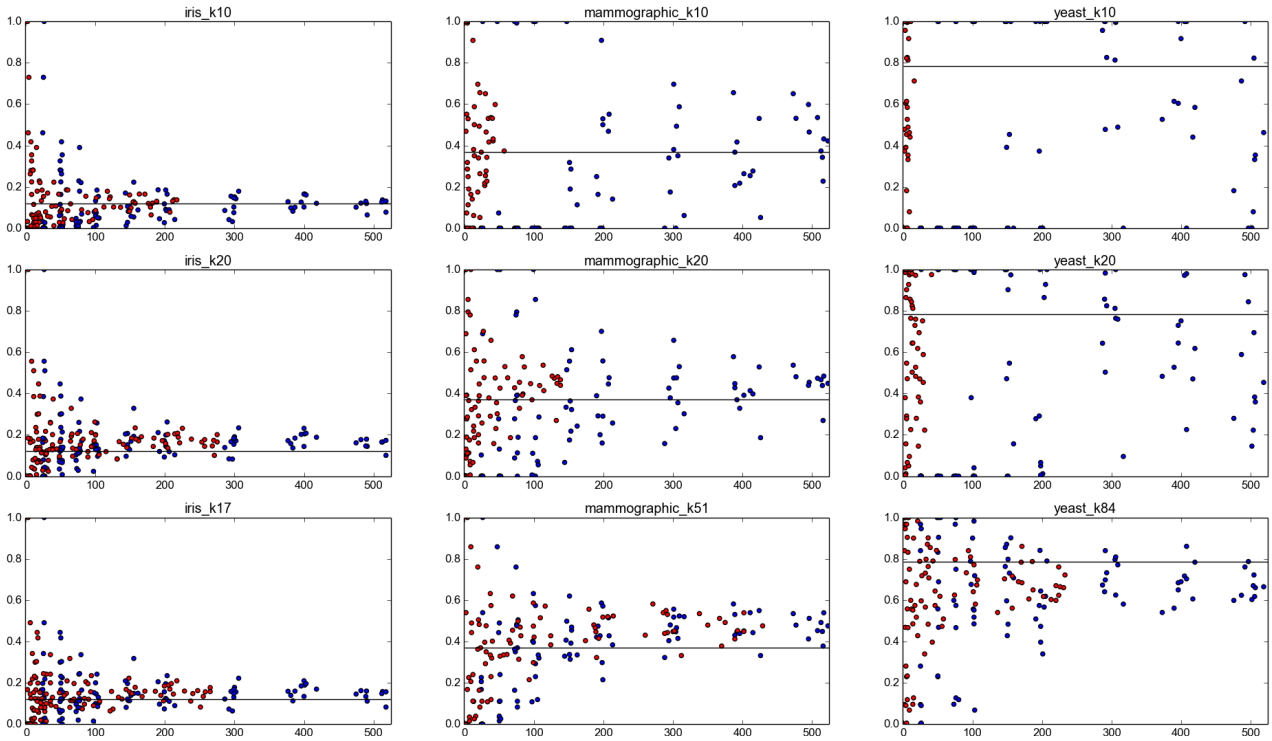


Figure 5. Convergence of the CBO w.r.t. the size of the constraint set. Three datasets are considered with increasing number of instances from left to right: iris(N=150), mammographic(N=830), yeast(N=1484). For each datasets, 80 constraint sets are sampled with various size (around 25,50,75,100,200,300,400,500). The CBO is computed for k=10 (top row), k=20 (middle row), k=10+N/20 (bottom row). The blue points correspond to the total number of constraints. The red points correspond to the number of constraints that actually participated in the measure. The Rvalue of the dataset (k=10, $\theta = 1$) is plotted as a black horizontal line.

ence that stability: the k-nearest neighbours to consider, the size of the dataset and the size of the constraint set. If the k is too small and the dataset too big, the measure would require too many constraints not to be noisy. To solve this problem, we need k to increase with the size of the dataset. For that reason, we set $k = 10 + N/20$ where N is the number of instances in the dataset. This has the desired effect while ensuring a minimal number of neighbours are considered for smaller datasets.

Figure 5 shows the variance of the CBO w.r.t. the size of the constraint set for 3 datasets of different sizes. For each dataset, several constraint sets of different sizes were sampled from the true labels. This shows that having a k increasing with the size of the dataset makes the CBO more stable.

4.3. Evaluation

The CBO is intended to serve as an alternative for the Rvalue, when the clusters are not known but some constraints are available. We therefore evaluate the CBO by comparing it to the Rvalue on a number of datasets from the UCI repository and the OpenML repository, namely *iris*, *glass*, *ionosphere*, *wine*, *vertebral*, *ecoli*, *seeds*, *students*, *robotnav4*, *yeast*, *zoo*, *breast cancer wisconsin*, *mammographic*, *banknote*, *haberman*, *segmentation*, *landsat*, *sonar*, *libras*, *hillvalley*, *optdigits*, *steel*, *leaf*, *spambase*, *parkinsons*, *occupancy*, *balance*, *pageblocks*, *diabetes*, *vehicle*, *authorship*, *ailerons*, *jedit*, *kc1*, *megawatt*, *blood*, *climate*, *fertility*, *heart*, *robotfail*, *volcanoes*, *engine*. For each dataset, 20 constraint sets of 200 random constraints were sampled. Figure 6 visualises how the Rvalue and the CBO (averaged over the 20 constraint sets) correlate, over the different datasets. This graph was produced for one particular value of k and θ for Rvalue, but other values give very similar results. With a correlation of 0.93, it is clear that CBO is useful as a proxy for Rvalue.

5. Algorithm selection

Now that we have the CBO to estimate overlap using constraints, we can use it for meta-learning, and more specifically algorithm selection. We picked 2 algorithms to select from: Expectation Maximization(EM) and Spectral Clustering(SC). We chose these two because among algorithms that build a global model like EM and algorithms that use local properties of the data like SC, these are 2 algorithms that perform the best on our datasets. To determine the performance of EM and SC, we ran the algorithms with different parameters and kept the best run. Then, we build 3 algorithm selection systems:

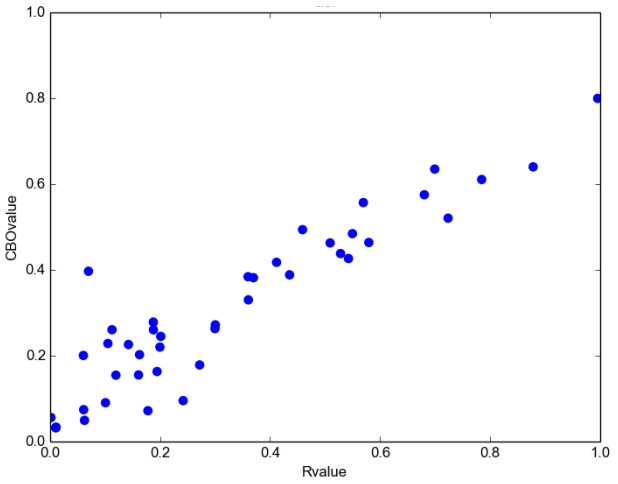


Figure 6. CBO with k=10+N/20 vs Rvalue with k=6 and θ=1.

- CBO: The first system only uses the CBO as meta-feature and choose EM if it is lower than 0.1, SC otherwise.
- Unsup: The second system uses unsupervised features that have been used by previous clustering algorithm selection system, and that are presented in table 3. As in (Ferrari & de Castro, 2015), we consider an attribute discrete if the number of distinct values observed for it is less than 30% of the number of instances. Using these meta-features, we learn a classifier to predict which of EM or SC will perform better.
- Full: The third system combines the unsupervised features and the CBO.

Unsupervised meta-feature description
Natural log of the number of instances
Natural log of the number of attributes
Percentage of outliers
Percentage of discrete attributes
Mean entropy of discrete attributes
Mean absolute correlation between distrete attributes
Mean skewness of continuous attributes
Mean kurtosis of continuous attributes
Mean absolute correlation between numerical attributes

Table 3. Unsupervised meta-features used for algorithm selection.

These 3 methods were run on the datasets presented in the previous section. For the constraint-based features, 20 constraint sets of about 200 constraints were sampled at random for each dataset. Table 5 shows the ARI averaged over datasets and constraint sets, using a leave-one-out cross validation for the 2 methods that

involved a classifier (Unsup and Ful). For those two methods, we used 3 classifiers: Support Vector Machine (SVM), Logistic Regression (LR) and Decision Trees (DT). For all algorithms (clustering, classifier, scores), we used the scikit-learn Python package (Pedregosa et al., 2011).

Classif.	EM	SC	CBO	Unsup	Full	Oracle
	0.31	0.32	0.33			0.37
SVM				0.33	0.33	
LR				0.33	0.33	
DT				0.33	0.31	

Table 4. Average ARI of multiple approaches: consistently EM or SC, selecting one of these using CBO, unsupervised features, or both ("Full"); and using an oracle to predict the best system.

Classif.	EM	SC	CBO	Unsup	Full	Oracle
	0.45	0.47	0.48			0.53
SVM				0.46	0.46	
LR				0.48	0.48	
DT				0.47	0.47	

Table 5. Same for datasets where either EM or SC scored an ARI of at least 0.2.

On average, algorithm selection methods perform a bit better than each algorithm separately. The improvement is quite modest, but relative to the maximum improvement possible (by using an oracle), still substantial. Interestingly, the CBO on its own performs as well as the whole set of features defined before. On the other hand, combining the CBO with those features does not further improve the results.

The choice of a threshold for the CBO method is rather flexible. We set it to 0.1 as it is a good value without being over-fitting. Figure 7 shows the variation of the performance of that method when varying that threshold for dataset with an ARI of at least 0.2 (which corresponds to the first line of table 5). It can be seen that any value between 0.1 and 0.3 has about the same score.

6. Conclusion

Algorithm selection and meta-learning have been studied mostly in the classification setting. In this paper, we have studied them in the context of clustering. Our main contributions are as follows.

First, we have identified *overlap between clusters* as a relevant property of the true clustering, meaning the clustering according to the true labels.

Second, because such overlap is difficult to quantify without knowing the cluster labels, we have proposed

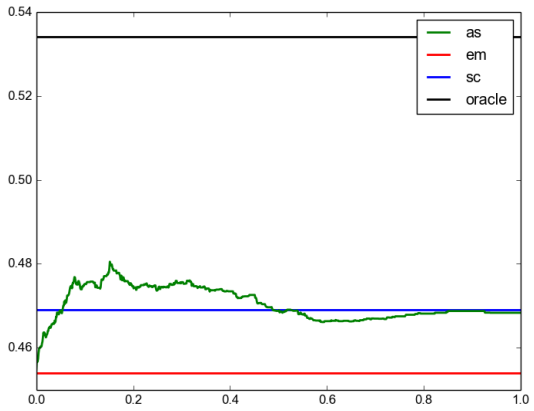


Figure 7. Performance of the CBO algorithm selection (AS) when the threshold for choosing EM or SC varies from 0 to 1 on the x axis.

a measure called CBO, which uses information from must-link / cannot-link constraints to estimate the amount of overlap. We have shown that the CBO correlates well with the Rvalue, a previously proposed measure for overlap in a completely known clustering. As such, the CBO can be a useful measure in itself, also outside the context of algorithm selection for clustering.

Third, we have empirically estimated the usefulness of selecting the most appropriate clustering method, among two methods with quite different properties: EM, which is good at detecting overlapping clusters but finds only elliptic clusters, and SC, which can find clusters of any shape but cannot return overlapping clusters. The conclusion is that the CBO is indeed informative for selecting the best among these two; it yields a small but noticeable improvement, and this improvement is comparable to the improvement obtained by using a set of 10 unsupervised features previously proposed for clustering algorithm selection. When combined with those other features, however, the CBO does not yield a further improvement. This suggests that the information contained in the CBO is already contained in the other features.

Compared to choosing the best clustering method using an oracle, CBO-based selection leaves room for further improvement. This is perhaps not surprising, given that the amount of overlap among clusters is one aspect that determines the effectiveness of clustering methods, but certainly not the only one. An indication of cluster shapes, for instance, is likely to give additional information. The question remains open to which extent this and other features can be derived

from constraints, and to what extent this can lead to better clustering algorithm selection.

Acknowledgments

Research financed by the KU Leuven Research Council through project IDO/10/012.

References

Adam, A., & Blockeel, H. (2015). Dealing with overlapping clustering: a constraint-based approach to algorithm selection. *Meta-learning and Algorithm Selection workshop-ECMLPKDD2015* (pp. 43–54).

Babaki, B., Guns, T., & Nijssen, S. (2014). Constrained clustering using column generation. In *Integration of ai and or techniques in constraint programming*, 438–454. Springer.

Bar-Hillel, A., Hertz, T., Shental, N., & Weinshall, D. (2005). Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, *6*, 937–965.

Bilenko, M., Basu, S., & Mooney, R. J. (2004). Integrating constraints and metric learning in semi-supervised clustering. *Proceedings of the twenty-first international conference on Machine learning* (p. 11).

De Souto, M. C., Prudencio, R. B., Soares, R. G., De Araujo, R. G., Costa, I. G., Ludermir, T. B., Schliep, A., et al. (2008). Ranking and selecting clustering algorithms using a meta-learning approach. *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on* (pp. 3729–3735).

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, 1–38.

Duong, K.-C., Vrain, C., et al. (2015). Constrained clustering by constraint programming. *Artificial Intelligence*.

Ferrari, D. G., & de Castro, L. N. (2015). Clustering algorithm selection by meta-learning systems: A new distance-based problem characterization and ranking combination methods. *Information Sciences*, *301*, 181–194.

Hu, P., Vens, C., Verstrynge, B., & Blockeel, H. (2013). Generalizing from example clusters. *Discovery Science* (pp. 64–78).

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of classification*, *2*, 193–218.

Oh, S. (2011). A new dataset evaluation method based on category overlap. *Computers in Biology and Medicine*, *41*, 115–122.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Pelleg, D., & Baras, D. (2007). K-means with large and noisy constraint sets. In *Machine learning: Ecml 2007*, 674–682. Springer.

Prudêncio, R. B., De Souto, M. C., & Ludermir, T. B. (2011). Selecting machine learning algorithms using the ranking meta-learning approach. In *Meta-learning in computational intelligence*, 225–243. Springer.

Ruiz, C., Spiliopoulou, M., & Menasalvas, E. (2007). C-dbscan: Density-based clustering with constraints. In *Rough sets, fuzzy sets, data mining and granular computing*, 216–223. Springer.

Soares, R. G., Ludermir, T. B., & De Carvalho, F. A. (2009). An analysis of meta-learning techniques for ranking clustering algorithms applied to artificial data. In *Artificial neural networks–icann 2009*, 131–140. Springer.

Van Craenendonck, T., & Blockeel, H. (2016). Constraint-based clustering selection. *arXiv preprint arXiv:1609.07272*.

von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, *17*, 395–416.

Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S., et al. (2001). Constrained k-means clustering with background knowledge. *ICML* (pp. 577–584).

Wang, X., & Davidson, I. (2010). Flexible constrained spectral clustering. *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 563–572).